Санкт-Петербургский государственный университет
Кафедра математического моделирования энергетических систем

Шувалов Денис Валерьевич

Магистерская диссертация

Применение эвристических алгоритмов в задаче определения местоположения ферромагнитных объектов

Направление 01.04.02
Процессы управления
Магистерская программа «Математическое и информационное обеспечение экономической деятельности»

Научный руководитель,
кандидат физ.-мат. наук,
доцент
Ю. Е. Балыкина

Санкт-Петербург
2017

Оглавление

ВВЕДЕНИЕ

Для обнаружения затонувших кораблей и подводных лодок (т. е. скрытых ферромагнитных объектов) используются разные методы, один из которых — метод обнаружения магнитной аномалии (magnetic anomaly detection). Каждый ферромагнитный объект создает аномалию в магнитном поле Земли из-за его высокой магнитной проницаемости по сравнению с неферромагнитными объектами. Обнаружение этой аномалии в магнитном поле Земли позволяет обнаружить цель. В отличие от активных методов обнаружения, метод обнаружения магнитной аномалии является пассивным методом, т. е. цель «не знает» о факте обнаружения. Во многих приложениях после обнаружения цели часто требуется определение местоположения цели и определение её магнитного момента. Метод обнаружения магнитной аномалии и определения местопололожения ферромагнитных объектов используется в таких приложениях, как обнаружение неразорвавшихся мин [1], затонувших кораблей или мест подводных крушений [2], обнаружения подводных лодок [3, 4] и надводных кораблей [5], в системах управления дорожным движением для того, чтобы определить наличие транспортных средств [6], для обнаружения скрытых ферромагнитных объектов у людей, проходящих через металлодетекторы (security check point systems) [7].

Летательные аппараты (самолеты и вертолеты) являются наиболее предпочтительными объектами для установки магнитометра, т. к. они быстро покрывают большие пространства. Эффективность метода обнаружения магнитной аномалии (ОМА) зависит от уровня шума принимаемого магнитометром сигнала. Поэтому необходимо компенсировать шум перед тем как решать задачу определения местоположения (локализации). Для оценки параметров шума обычно используются следующие методы: метод наименьших квадратов [8], фильтры с конечной импульсной характеристикой [9] и модель малого сигнала [10].

В работе рассматривается трехкомпонентный магнитометр, зафиксированный на неподвижном носителе и для подвижной цели предполагается модель магнитного диполя.

ПОСТАНОВКА ЗАДАЧИ

Для описания сигнала магнитной аномалии, порождаемой ферромагнитной целью, используется модель диполя:

$$\mathbf{B}(\mathbf{m}, \mathbf{r}) = \frac{\mu_0}{4\pi}\left[\frac{3(\mathbf{m}, \mathbf{r})\mathbf{r}}{|\mathbf{r}|^5} - \frac{\mathbf{m}}{|\mathbf{r}|^3}\right], \tag{1}$$

где $\mathbf{B}(\mathbf{m}, \mathbf{r})$ — магнитное поле, порождаемое объектом на расстоянии $\mathbf{r}$ от объекта (цели), $\mathbf{m}$ — вектор магнитного момента цели, а μ_0 — проницаемость свободного пространства (известная константа). Объединяя магнитное поле цели вместе с магнитным полем Земли получаем магнитное поле, измеряемое магнитометром.

$$\mathbf{B}_M = \mathbf{B}(\mathbf{m}, \mathbf{r}) + \mathbf{B}_E, \tag{2}$$

$\mathbf{B}_M$ — измеренное магнитное поле в точке $\mathbf{r}$ от цели, $\mathbf{B}_E$ — магнитное поле Земли. Таким образом, магнитометр, считывающий магнитное поле в моменты времени $t_1, \cdots, t_j, \cdots, t_N$, и, находящийся на расстоянии $\mathbf{r}_1, \cdots, \mathbf{r}_j, \cdots, \mathbf{r}_N$ в эти моменты времени, выдает выборку $B_M^1, \cdots, B_M^j, \cdots B_M^N$, где N — объем выборки, $B_M^j \in \mathbb{R}^3$. Тогда, получаем нелинейную переопределённую систему:

$$\begin{cases} \mathbf{B}_M^j = \dfrac{\mu_0}{4\pi}\left[\dfrac{3(\mathbf{m}, \mathbf{r}_1)\mathbf{r_1}}{|\mathbf{r}_1|^5} - \dfrac{\mathbf{m}}{|\mathbf{r}_1|^3}\right] + \mathbf{B}_E, \\ \mathbf{B}_M^2 = \dfrac{\mu_0}{4\pi}\left[\dfrac{3(\mathbf{m}, \mathbf{r}_2)\mathbf{r_2}}{|\mathbf{r}_2|^5} - \dfrac{\mathbf{m}}{|\mathbf{r}_2|^3}\right] + \mathbf{B}_E, \\ \cdots \\ \mathbf{B}_M^N = \dfrac{\mu_0}{4\pi}\left[\dfrac{3(\mathbf{m}, \mathbf{r}_N)\mathbf{r_N}}{|\mathbf{r}_N|^5} - \dfrac{\mathbf{m}}{|\mathbf{r}_N|^3}\right] + \mathbf{B}_E \end{cases} \tag{3}$$

в предположении что расстояние $\mathbf{r}_1, \cdots, \mathbf{r}_N$ выражается через единственный вектор $\mathbf{r}$ (зависит от типа движения магнитометра и цели).

Таким образом, требуется решить систему (3) относительно шести параметров $\mathbf{r}, \mathbf{m}$, т.е. появляется задача оптимизации:

$$\min \mathbf{f}(\mathbf{r}, \mathbf{m}) = \sum_{j=1}^{N} \|\mathbf{B}_M^j - \mathbf{B}^j(\mathbf{m}, \mathbf{r}) + \mathbf{B}_E\|.$$

Работа заключается в исследовании эвристических методов оптимизации для решения задачи определения местоположения ферромагнитных объектов, а также в сравнении эффективности данных методов между собой. В качестве параметров сравнения эффективности используются точность решения, скорость сходимости и время выполнения программы.

Глава 1

Теоретические сведения

Основным свойством ферромагнетиков является способность данных веществ обладать намагниченностью при отсутствии внешнего магнитного поля. Намагниченный объект порождает магнитное поле, которое может быть обнаружено магнитометром. Магнитное поле объекта может быть аппроксимировано полем диполя при измерениях, проводимых на достаточном расстоянии от объекта. Магнитное поле $\mathbf{B}_M(\mathbf{m}, \mathbf{r})$ измеренное магнитометром равно сумме магнитного поля объекта и магнитного поля Земли $\mathbf{B}_E$, которое по предположению известно и равно константе.

$$\mathbf{B}_M(\mathbf{m}, \mathbf{r}) = \frac{\mu_0}{4\pi}\left[\frac{3(\mathbf{m}, \mathbf{r})\mathbf{r}}{|\mathbf{r}|^5} - \frac{\mathbf{m}}{|\mathbf{r}|^3}\right] + \mathbf{B}_E, \tag{1.1}$$

где вектор $\mathbf{B}_M(\mathbf{m}, \mathbf{r})$ — измеренное магнитное поле в точке $\mathbf{r}$ от цели, $\mathbf{m}$ — вектор магнитного момента цели, а μ_0 — проницаемость свободного пространства.

В различных источниках рассматриваются разные модели взаимного движения цели и платформы с магнитометром. Рассмотрим две следующих модели. Предположения первой модели заключаются в том, что цель неподвижна, а платформа с магнитометром совершает некоторое движение в поисках цели, которое не обязательно является прямолинейным. Будем называть такую модель — моделью с неподвижной целью. Предположения второй модели наоборот заключаются в том, что цель совершает равномерное прямолинейное движение, а платформа с магнитометром неподвижна.

Будем называть такую модель — моделью с подвижной целью.

1.1 Модель с неподвижной целью

Рассмотрим случай, при котором цель неподвижна, а летательный аппарат с расположенным на нем магнитометром совершает движение в пространстве в поисках цели, см. рис. 1.1. На протяжении полета магнитометр измеряет магнитное поле.

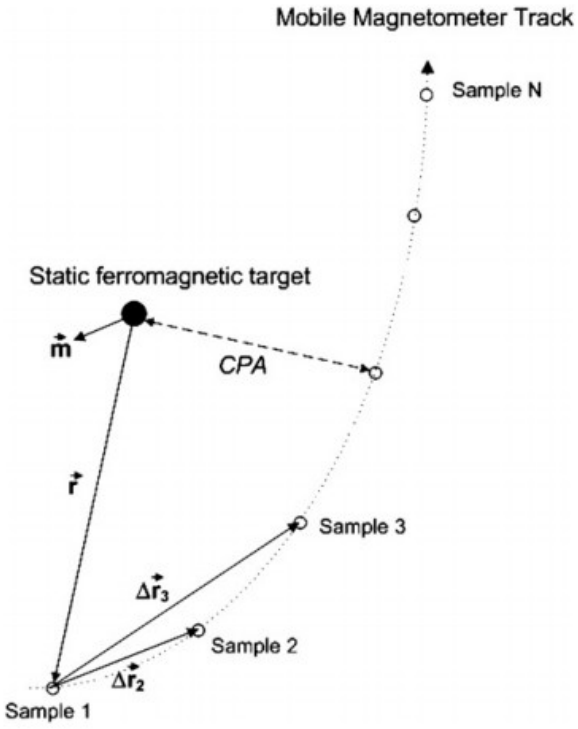

Рис. 1.1: Установленный на подвижную платформу векторный магнитометр в поисках цели [27].

Рассмотрим выборку объема N измеренного магнитометром магнитного поля. Для $N >> 1$ получаем нелинейную переопределённую систему:

$$\begin{cases} \mathbf{B}_M^j = \dfrac{\mu_0}{4\pi}\left[\dfrac{3(\mathbf{m},\mathbf{r}_1)\mathbf{r_1}}{|\mathbf{r}_1|^5} - \dfrac{\mathbf{m}}{|\mathbf{r}_1|^3}\right] + \mathbf{B}_E, \\ \mathbf{B}_M^2 = \dfrac{\mu_0}{4\pi}\left[\dfrac{3(\mathbf{m},\mathbf{r}_2)\mathbf{r_2}}{|\mathbf{r}_2|^5} - \dfrac{\mathbf{m}}{|\mathbf{r}_2|^3}\right] + \mathbf{B}_E, \\ \quad \cdots \\ \mathbf{B}_M^N = \dfrac{\mu_0}{4\pi}\left[\dfrac{3(\mathbf{m},\mathbf{r}_N)\mathbf{r_N}}{|\mathbf{r}_N|^5} - \dfrac{\mathbf{m}}{|\mathbf{r}_N|^3}\right] + \mathbf{B}_E \end{cases} \tag{1.2}$$

Местоположение магнитометра в каждый момент времени считется известным, поэтому определены расстояния между точками местоположения $\Delta\mathbf{r}_2$, $\Delta\mathbf{r}_3$, $\cdots$, $\Delta\mathbf{r}_N$. Вектор из каждой точки местоположения магнитометра к цели вычисляется следующим образом.

$$\mathbf{r}_i = \mathbf{r} + \Delta\mathbf{r}_i,\ i = 2, \cdots, N. \tag{1.3}$$

Таким образом, требуется решить систему (1.2) относительно шести неизвестных $\mathbf{r}$ и $\mathbf{m}$ — вектора расстояния между целью и первой точкой и вектора магнитного момента цели соответственно. Аналитическое решение данной задачи получить достаточно сложно, особенно в условиях наличия шума. Поэтому дискретный подход может оказаться более эффективным. Для каждой переменной векторов $\mathbf{r}$ и $\mathbf{m}$ необходимо задать диапазон допустимых значений и указать требуемую точность. Для пояснения рассмотрим следующий пример. Пусть требуется найти затонувший корабль магнитный момент которого в диапазоне от -100000 А·м^2 до 100000 А·м^2 и требуемая точность магнитного поля 2000 А·м^2. Полагая что поиски корабля ведутся в кубе со стороной 1000 м и требуемая точность местоположения — 10 м, получаем, что каждой переменной соответствует 100 возможных состояний. Поэтому для шести переменных система имеет 100^6 различных состояний, из которых верное только одно, которое ближе всех находится к действительным значениям (например, в смысле евклидовой метрики). Таким образом непрерывная система уравнений сводится к решению дискретной системы. Это приводит к поиску решений с заданной точностью, но не к поиску оптимального решения, что для некоторых приложений может быть вполне достаточно. Перебор всех состояний системы

занял бы достаточно много времени что неприменимо к системам реального времени, поэтому предлагаются эвристические методы для решения данной задачи.

1.2 Модель с подвижной целью

Рассмотрим экспериментальную установку, состоящую из двух трехкомпонентных магнитометров, зафиксированных на земле. Требуется определить координаты цели и её магнитный момент, всего шесть неизвестных параметров. Введем обозначения (см. рис. 1.2): α — угол пересечения (между траекторией движения цели и линией, соединяющей два датчика), d — дистанция пересечения (от точки пересечения до одного из датчиков), $\mathbf{V}$ — вектор скорости движения цели, $\mathbf{M}$ — вектор магнитного момента цели и $\mathbf{R} = \mathbf{R}(t)$ — расстояние между движущейся целью и первым магнитометром.

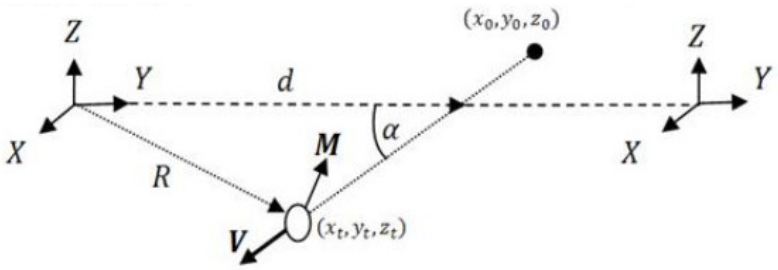

Рис. 1.2: Экспериментальная установка, содержащая два векторных магнитометра.

Предполагается, что цель свободно перемещается рядом с магнитометрами и среда измерения не загрязнена магнитным шумом и помехами. Цель имеет постоянную скорость движения (обычно 1 м/с) и может пересекать линию, соединяющую датчики, в любой точке и под любым углом.

Глава 2

Обзор научных публикаций

Метод обнаружения магнитной аномалии (ОМА) сводится к задаче оптимизации, для решения которых будем рассматривать численные и эвристические методы. Многие стандартные методы оптимизации не подходят для определения местоположения ферромагнитных объектов из-за большого отношения сигнал/помеха или из-за времени, которое требуется для сходимости алгоритма.

Для подавления шума и улучшения отношения сигнал/помеха применяются метод подавления когерентного шума (adaptive coherent noise suppression) [11], вейвлет-преобразование [12].

2.1 Численные методы

Из численных методов используются модификации метода Ньютона для нахождения корня функции: метод Ньютона-Рафсона, Гаусса-Ньютона и наиболее распространенный алгоритм Левенберга-Маркварда для решения задачи о наименьших квадратах [13, 14]. Главное преимущество алгоритма Левенберга-Маркварда — это быстрая сходимость. Однако для достаточно хорошего решения требуется относительно хорошее начальное приближение и отношение сигнал/помеха не меньше пяти. Также применяется алгоритм Левинсона-Дарбина как альтернатива быстрому преобразованию Фурье [15].

2.2 Эвристические методы

Среди эвристических методов для определения местоположения может быть использованы следующие алгоритмы и методы:

1. Генетический алгоритм[2, 16], который применим даже при отношении сигнал/помеха близкому к единице [17].

2. Метод отжига [18, 19].

3. Муравьиный алгоритм, который, не смотря на его сложную реализацию, требует достаточно много времени для сходимости и низкую точность решения [17].

4. Алгоритм гравитационного поиска [20].

5. Метод роя частиц [21, 22].

В системах обнаружения магнитной аномалии используются разные подходы в зависимости от взаимного движения цели и носителя с сенсором (магнитометром) [23]. В первом подходе, при известном взаимном движении цели и сенсора, например, когда цель неподвижна, а движение носителя заложено заранее и его местоположение непрерывно измеряется системой определения местоположения. Этот подход часто основан на предположении, что ферромагнитная цель является точкой магнитного диполя [24]. Во втором подходе алгоритм обнаружения магнитной аномалии применяется, когда нет никаких предположений о взаимном движении цели и сенсора. В данном подходе могут использоваться методы, основанные на статистическом анализе полученного шума.

2.2.1 Генетический алгоритм

Генетический алгоритм, адаптированный под задачу обнаружения магнитной аномалии рассмотрен в [17]. Алгоритм используется в таких приложениях как анализ данных, электротехника, проектировании систем управления и многих других.

Предварительная обработка данных. Для улучшения эффективности и сходимости алгоритма данные предварительно обрабатываются: применяется фильтрация сигнала и исключение тренда. Т. е. вначале применяется фильтр нижних частот: низкочастотное дискретное косинусное преобразование [25], которое сглаживает данные и эффективно удаляет высокочастотный шум. Затем применяется фильтр верхних частот Ходрика-Престотта [26].

Моделирование начальной популяция. Случайным образом создается начальная популяция состоящая из набора хромосом. Каждая хромосома в свою очередь состоит из шести генов: α, d, v и трех компонент вектора магнитного момента цели $\mathbf{M}$. Для каждого гена начальное значение выбирается как равномерно распределенное в области определения гена. Каждой хромосоме единственным образом можно сопоставить траекторию движения цели и вычислить магнитное поле

$$\mathbf{B}(\mathbf{m}, \mathbf{r}) = \frac{\mu_0}{4\pi}\left[\frac{3(\mathbf{m}, \mathbf{r})\mathbf{r}}{|\mathbf{r}|^5} - \frac{\mathbf{m}}{|\mathbf{r}|^3}\right]. \tag{2.1}$$

где $\mathbf{r} = (x, y, z)^T$, $x = x(t) - x_s$, $y = y(t) - y_s$, $z = z(t) - z_s$, величины x_s, y_s, z_s — координаты сенсора, R — модуль вектора (x, y, z), т. е. расстояние от сенсора до цели.

Выбор функции приспособленности. Она используется для того, чтобы определить, насколько вычисленный сигнал близок к сигналу, записанному магнитометром. Функция должна удовлетворять двум критериям. Она должна быть в меру чувствительной, т. е. должна иметь не вертикальный градиент, и в то же время ненулевой. Также её вычисление не должно быть затратным, т. к. вычисление данной функции производится много раз и значительно влияет на эффективность алгоритма. В качестве такой функции можно рассматривать расстояние Фреше и расстояние Хаусдорфа. Первое имеет более строгое определение, но и также имеет большую вычислительную сложность. Другим более быстрым способом определения сходства функций является вычисление расстояния с использованием

функции взаимной корреляции, которая заключается в скалярном произведении двух функций. Эффективность использования расстояния Хаусдорфа и взаимно корреляционной функции рассматривается в [17].

Для удобства производится нормировка скалярного произведения, чтобы областью значения функции являлся отрезок $[0, 1]$. Взаимно корреляционная функция двух временных рядов P и Q выглядит следующим образом:

$$f_{cc} = \frac{(P \cdot Q)}{max\{P \cdot P, Q \cdot Q\}} \tag{2.2}$$

где $\cdot$ обозначает скалярное произведение.

$$P \cdot Q = \sum_{i=1}^{length(P)} P_i Q_i. \tag{2.3}$$

Главный цикл. Когда смоделирована начальная популяция, выполняются следующие шаги.

(1) Расчёт функции приспособленности. Для каждой хромосомы вычисляется соответствующая геометрическая траектория и сигнал (2.1). Вычисляется функция приспособленности (2.2) смоделированного и измеренного сигнала, которая показывает "расстояние"между вычисленным сигналом $\mathbf{B}^j$ и записанным сигналом $\mathbf{B}$.

Далее применяются две нестандартные процедуры. Первая заключается в том, что десять хромосом с наилучшим значением функции приспособленности отделяются от остальной популяции и попадают в следующее поколение. Также вычисляется величина $\tilde{f}$ — значение наилучшей функции приспособленности как среднее арифметическое десяти лучших хромосом. Вторая процедура заключается в корректировке размера популяции на каждом шаге. Для начальных популяций требуется большое количество особей для того, чтобы полностью исследовать область определения. Но размер популяции значимо влияет на скорость работы алгоритма, по-

этому размер популяции в последующих поколениях должен быть уменьшен.

После генерации хотя бы десяти поколений начинаем проверять критерий останова. Алгоритм завершает работу если выполняется одно из следующих условий:

- Текущая итерация n превышает максимально допустимое число N_{max}.
- Наилучшее значение функции приспособленности $\tilde{f}$ превышает некоторый порог. Обычно это значение между 0.85 и 0.9.
- Наилучшее значение функции приспособленности приспособленности $\tilde{f}$ не меняется (значимо) в течении последних десяти поколений. Наилучшее изменение $d\tilde{f}(n)$ вычисляется согласно формуле

$$d\tilde{f}(n) = 100\frac{\sum_{j=n-9}^{n}\left[\tilde{f}(j) - \tilde{f}(j-1)\right]}{\sum_{j=n-9}^{n}\left[\tilde{f}(j)\right]}. \tag{2.4}$$

  Если $d\tilde{f}(n)$ меньше некоторого заданного значения, то алгоритм завершает работу.

(2) Селекция. Производится выборка хромосом согласно вероятности, пропорциональной их значениям приспособленности. Десять хромосом, отделенные на предыдущем шаге также участвуют в селекции. Каждой хромосоме сопоставляется отрезок, длина которого пропорциональна её значению функции приспособленности. Эти отрезки совмещаются и нормируются. Применяем экспоненциальное масштабирование, которое на начальных поколениях делает хромосомы более равными, а на конечных поколениях выбираются преимущественно наиболее приспособленные хромосомы.

$$f_s = f_{cc}^{k(n)}, \tag{2.5}$$

где

$$k(n) = \left(tan\left[\frac{\pi n}{2(N_{max}+1)}\right]\right)^{0.1}. \tag{2.6}$$

(3) Скрещивание. Для каждой пары хромосом, участвующих в скрещивании (от 70 до 90%) выбирается точка среза в цепочке гена. Точка разреза отделяет «голову» и «хвост». И две хромосомы меняются «головами» и образуют две новые хромосомы. Если у получившихся хромосом значения функции приспособленности больше, чем у породивших их хромосом, то они замещают своих «родителей». Если же у хромосом «плохие» значения функции приспособленности, то они не участвуют в дальнейшем.

(4) Мутация. Мутация позволяет генетическому алгоритму достигнуть любой точки области поиска без применения процедуры селекции и скрещивания. Хромосомы для мутации выбираются используя гипергеометрическое распределение. Выбираются от 5 до 20% особей из популяции. Для каждой из этих хромосом один или более мутирующих генов выбираются случайным образом. Новое значение гена получается добавлением гауссового шума к исходному значению.

(5) Расселение. Десять лучших хромосом, на которые не воздействовали скрещиванием и мутацией, замещают в популяции десять наихудших хромосом.

(6) Переход к шагу (1).

Параметры алгоритма:

- N_{max} — максимальное количество итераций;
- T — начальная температура;
- k — коэффициент уменьшения температуры, $0 < k < 1$;
- N_{neig} — максимальное количество неудачно генерируемых состояний перед переходом на следующую итерацию.

2.2.2 Метод отжига

Метод отжига (алгоритм Метрополиса) является вероятностным методом для решения задачи поиска глобального экстремума некоторой целевой функции имеющей несколько локальных экстремумов. В основе метода лежит имитация физического процесса — кристализации тела.

В предположении модели описанной в разделе 1.1 задача оценки характеристик цели сводится к решению нелинейной и переопределённой системы уравнений. Компьютерное моделирование демонстрирует высокую точность решения и малые затраты времени при наличии шума высоких уровней [27]. Данный метод эффективнее метода возрастающего обучения на основе популяции (population-based incremental learning) (в плане отношения сигнал/помеха) при решении практических задач.

Алгоритм метода отжига. Предположим, что система может находиться в состояниях $\{S_1, \dots, S_j, \dots\}$, где $S_j = (m_x^j, m_y^j, m_z^j, r_x^j, r_y^j, r_z^j)$. Каждый параметр определен на некотором ограниченном промежутке. Состоянию S_j сопоставляется энергия

$$E_j = \frac{1}{N} \sum_{j=1}^{N} \|B^j - B_c^j(S_j)\|^2. \tag{2.7}$$

Чем ближе энергия состояния к нулю, тем ближе состояние к точке глобального минимума энергетической функции.

Вероятность перехода из состояния S_j в некоторое состояние S_k рассчитывается следующим образом:

$$p_{jk} = \begin{cases} 1, & \text{если } E_k \le E_j, \\ exp\big(\frac{E_j - E_k}{T}\big), & \text{если } E_k > E_j, \end{cases} \tag{2.8}$$

где — параметр температуры. Обычно значение температуры уменьшается с каждым переходом из одного состояния в другое. Большое значение температуры соответствует большой вероятности избежать локального

минимума, в то время как маленькое значение температуры на последних итерациях позволяет приблизиться к глобальному минимуму.

В главном цикле алгоритма метода отжига каждая итерация состоит в следующем.

1. Сравнить энергию текущего состояния S_j с найденным на текущий момент глобальным минимумом.

2. Сгенерировать новое состояние S_k.

3. Вычислить значение E_k, и вероятность p_{jk}.

4. Сгенерировать случайное число α, равномерное распределённое на отрезке $[0, 1]$.

5. Если $\alpha < p_{jk}$, то текущему состоянию S_j присвоить состояние S_k, уменьшить температуру $T = kT$ и перейти к следующей итерации алгоритма. Если же $\alpha > p_{jk}$, то вернуться к шагу 2.

Параметры алгоритма:

- N_{max} — максимальное количество итераций;
- T — начальная температура;
- k — коэффициент уменьшения температуры, $0 < k < 1$;
- N_{neig} — максимальное количество неудачно генерируемых состояний перед переходом на следующую итерацию.

Также за параметр алгоритма можно считать функцию распределения генерируемого состояния, вычисление энергии, и функцию уменьшения температуры.

2.2.3 Метод роя частиц

Метод роя частиц (МРЧ) является одним из методов численной оптимизации, основанный на модели социального поведения группы животных, в которой отсутствует лидер. Обычно такая группа животных находит пищу случайным образом, следуя за одним из членов группы, который ближе всего находится к еде (к возможному решению). Животные общаются между собой, и поэтому те, кто находится дальше от источника пищи постепенно к нему приближаются. Это повторяется до тех пор, пока не найдется самый лучший источник пищи. Алгоритм МРЧ основан на данном процессе. Имеется некоторая область, в которой требуется найти глобальный экстремум (самый большой источник пищи). Рой состоит из частиц, каждая из которых представляет возможное решение (точки в пространстве поиска) и передвигается к источнику пищи (к экстремуму).

Для нахождение глобального экстремума требуется хорошее исследование всей области поиска, которое даст приблизительные решения экстремумов, а затем требуется сконцентрировать внимание в самых перспективных областях для нахождения более точного решения. В каждый момент времени частица помнит свое лучшее положение ($lb_i(t)$), а также помнит лучшее положение всего роя за все прошедшее время ($gb(t)$). Позиция частицы зависит от скорости. Пусть $x_i(t)$ — означает положение частицы в пространстве поиска во время t. В алгоритме рассматривается дискретное время. Позиция частицы меняется на каждой итерации алгоритма путем добавления скорости $v_i(t)$:

$$x_i(t+1) = x_i(t) + v_i(t+1), \tag{2.9}$$

где скорость частицы j задана уравнением:

$$v_i(t) = \omega v_i(t-1) + c_1 r_1 (lb_i(t) - x_i(t-1)) + c_2 r_2 (gb(t) - x_i(t-1))$$

и $x_i(0) \sim \mathbf{U}(x_{min}, x_{max})$ сгенерировано при помощи равномерного распределения в области поиска, c_1 и c_2 — коэффициенты ускорения, r_1 и r_2 —

случайные вектора. Пусть имеется целевая функция и заданы ограничения:

$$\max \mathbf{f}(x)$$
$$x(A) \leq x < x(B).$$

Алгоритм заключается в следующем.

1. Задать начальные параметры: количество итераций алгоритма N_max, количество частиц в рое N и величины ω, c_1, c_2.

2. Сгенерировать начальную популяцию (рой) $x_1(0), \cdots, x_n(0)$ с помощью равномерного распределения в области поиска. Вектор x_j называется частицой j или вектором координат частицы j.

3. Вычислить значение целевой функции $f(x_1(0)), \cdots, f(x_n(0))$.

4. Инициализировать скорость $v_j = 0$ и лучшее положение для каждой частицы $p_j = 0$ для $j = 1, \cdots, N$.

   Основной цикл. Для $t = 1, \cdots, N_max$ (либо другой критерий останова).

   (a) Запомнить лучшее положение каждой частицы и обозначить как p_j для $j = 1, \cdots, N$. Если значение целевой функции частицы x_j больше чем p_j, то $p_j = x_j(t)$. И запомнить лучшее глобальное положение среди всех частиц и обозначить как $g(t)$.

   (b) Вычислить скорость v_j частицы $x_j(t)$ по формуле:

   $$v_i(t) = \omega v_i(t-1) + c_1 r_1 (lb_i(t) - x_i(t-1)) + c_2 r_2 (gb(t) - x_i(t-1))$$

   Параметры c_1 и c_2 отражают степень влияния лучшей позиции частицы по отношению к лучшей позиции всего роя.

   (c) Вычисление новой позиции каждой частицы: $x_i(t+1) = x_i(t) + v_i(t+1)$.

5. Лучшее состояние хранится в переменной $g(t)$.

Для данного метода разработано некоторое количество модификаций с целью улучшить сходимость и качество решения [22].

2.2.4 Метод гравитационного поиска

Метод гравитационного поиска основан на законе гравитации. Все объекты притягивают друг друга посредством силы гравитации, тем самым порождая движение всей системы к самым тяжелым объектам. Тяжелые объекты, соответствующие хорошим решениям, двигаются медленно, что положительно сказывается в исследовании области. Каждому объекту соответствует положение в пространстве и некоторая масса. Положение объекта соответствует решению, а масса отражает насколько решение является близким к оптимальному.

В основе метода лежат два закона — закон гравитации и закон движения. Сила гравитации все время воздействует на все объекты. Сила гравитации между двумя частицами прямо пропорциональна произведению их масс и обратно пропорциональна квадрату расстояния между ними:

$$F = G\frac{M_1 M_2}{R^2},$$

где F — сила гравитационного притяжения, G — гравитационная постоянная, M_1 и M_2 — массы первой и второй частицы соответственно, R — расстояние между двумя частицами. Второй закон Ньютона гласит, что если сила F воздействует на частицу, то ее ускорение a зависит только от этой силы и от своей массы:

$$a = \frac{F}{M}.$$

Отметим, что из-за эффекта уменьшения гравитации фактическое значение гравитационной постоянной G зависит от возраста вселенной. Поэтому гравитационная постоянная — это убывающая функция, зависящая от времени:

$$G(t) = G(t_0) \cdot \left(\frac{t_0}{t}\right)^{\beta}, \ \beta < 1.$$

Алгоритм гравитационного поиска заключается в следующем.

1. Задать начальные параметры: количество итераций алгоритма N_{max}, количество частиц в системе N и начальное значение $G(t_0)$.

2. Сгенерировать начальную популяцию $x_1(0), \cdots, x_n(0)$ с помощью равномерного распределения в области поиска. Вектор $x_j = (x_j^1, \cdots, x_j^d)$ является вектором положения частицы j в d-мерном пространстве. Инициализировать вектор скоростей $v_i^d(0) = 0$ для $i = 1, \cdots, n$

   Основной цикл. Для $t = 1, \cdots, N_{max}$ (либо другой критерий останова).

   (a) Вычислить значение приспособленности для каждой частицы $fit_i(t)$, найти худшее (минимальное) и лучшее (максимальное) из них:
   $$w(t) = \min_{i=1}^{n} fit_i(t), \ b(t) = \max_{i=1}^{n} fit_i(t).$$

   (b) Вычислить массу каждой частицы $M_i(t)$:
   $$m_i(t) = \frac{fit_i(t) - w(t)}{b(t) - w(t)}, \ M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{n} m_i(t)}.$$

   (c) Вычислить силу гравитации, т.ė. воздействия частицы j на частицу i:
   $$F_{ij}^d = G(t)\frac{M_i M_j}{R_{ij} + \epsilon}(x_j^d(t) - x_i^d(t)),$$
   где ϵ — некоторая малая константа, R_{ij} —евклидово расстояние между частицами i и j. Использование в знаменателе R вместо R^2 дает лучшие результаты работы алгоритма.

   (d) Вычислить суммарную силу притяжения, воздействующую на частицу i:
   $$F_i^d(t) = \sum_{j=1, j\neq i}^{N} \alpha_j F_{ij}^d(t), \ i = 1, \cdots, n,$$

где α_j равномерно распределенное число на отрезке $[0, 1]$.

(e) Вычислить ускорение частицы:

$$a_i^d(t) = \frac{F_i^d(t)}{M_i^d(t)}.$$

(f) Вычислить новые скорости:

$$v_i^d(t+1) = \alpha_i v_i^d(t) + a_i^d(t).$$

(g) Вычислить новую позицию каждой частицы:

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1),$$

где α_j равномерно распределенное число на отрезке $[0, 1]$.

(h) Вычислить гравитационную постоянную $G(t) = G(G_0, t)$

Глава 3

Сравнение методов

3.1 Моделирование сигнала

Можно рассматривать разные постановки задач, которые описаны в разделах 1.1 и 1.2. В первой модели предполагается, что магнитометр расположен на подвижной платформе, цель является стационарной и в качестве параметров используется вектор из точки начала движения платформы к цели $\mathbf{r}$. Во второй модели рассматривается случай когда магнитометр установлен на стационарной платформе, цель совершает прямолинейное равномерное движение и в качестве параметров используются угол пересечения α, расстояние d и скорость v. Общим параметром для обеих моделей служит вектор магнитного момента $\mathbf{m}$.

Для сравнения эффективности работы различных методов выбрана модель с подвижной целью. Для моделирования сигнала достаточно задать шесть параметров модели: α, d, v, m_x, m_y, m_z. Для сравнения эффективности алгоритмов рассмотрим следующие параметры (таб 3.1) и их соответствующие ограничения (область поиска).

Зададим координаты первого магнитометра $(x_s, y_s, z_s)^T = [0, 0, 0]^T$, частоту сигнала 10 Гц, временной интервал движения платформы с магнитометром $[0, 200]$ с. По известным параметрам α, d, v рассчитаем траекторию движения $\mathbf{r} = (r_x, r_y, r_z)$ в горизонтальной плоскости, а затем смоделируем сигнал — магнитное поле объекта:

	Истинное значение	Размерность	Область определения
α	$\frac{\pi}{4}$	радиан	$[0, \pi]$
d	20	м	$[0, 40]$
v	2	м/с	$[0, 5]$
m_x	0.5	Am^2	$[-10, 10]$
m_y	0.1	Am^2	$[-10, 10]$
m_z	-0.2	Am^2	$[-10, 10]$

Таблица 3.1: Параметры сигнала цели.

$$\begin{cases} \mathbf{B}^j = \dfrac{\mu_0}{4\pi}\left[\dfrac{3(\mathbf{m}, \mathbf{r}_1)\mathbf{r_1}}{|\mathbf{r}_1|^5} - \dfrac{\mathbf{m}}{|\mathbf{r}_1|^3}\right], \\ \mathbf{B}^2 = \dfrac{\mu_0}{4\pi}\left[\dfrac{3(\mathbf{m}, \mathbf{r}_2)\mathbf{r_2}}{|\mathbf{r}_2|^5} - \dfrac{\mathbf{m}}{|\mathbf{r}_2|^3}\right], \\ \cdots \\ \mathbf{B}^N = \dfrac{\mu_0}{4\pi}\left[\dfrac{3(\mathbf{m}, \mathbf{r}_N)\mathbf{r_N}}{|\mathbf{r}_N|^5} - \dfrac{\mathbf{m}}{|\mathbf{r}_N|^3}\right], \end{cases} \tag{3.1}$$

которое будем измерять в нТл. К сигналу добавим гауссовский шум с дисперсией $\sigma^2 = 10^{-4}$ нТл. Получим следующий сигнал на рис. 3.1 с отношением сигнал/помеха $SNR \approx 5$ к которому прибавим магнитное поле Земли B_E.

Далее обозначим полученный сигнал как B_M и будем считать этот сигнал известным (измеренным магнитометром), а истинные значения параметров из таблицы 3.1 неизвестными переменными, оценки которых требуется найти.

Зададим область поиска $D \in \mathbb{R}^6$ согласно таблице 3.1 и будем рассматривать точки $x_i \in D$, где $x_i = (x_i^1, \cdots, x_i^6) = (\alpha, d, v, m_i^x, m_i^y, m_i^z)$. В качестве целевой функции $\mathbf{f}$ требуется задать такую функцию, которая отражает меру сходства измеренного сигнала B_M и сигнала B_J, который будет смоделирован для точки x_j. В качестве целевой функции рассмотрим

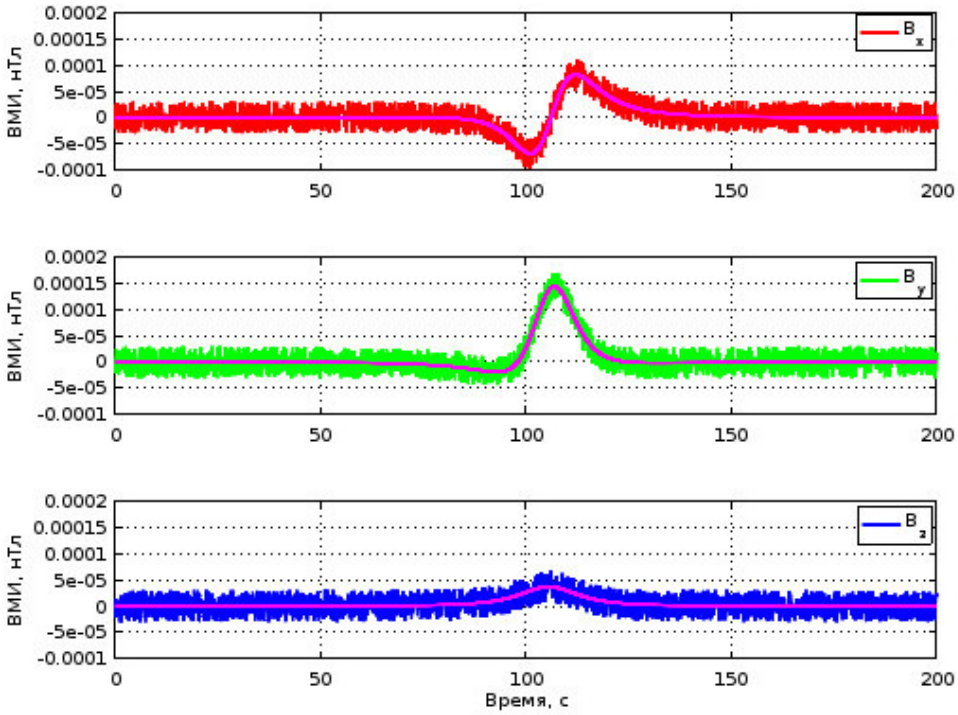

Рис. 3.1: Исходный сигнал магнитного поля цели и сигнал с шумом B.

взаимно корреляционную функцию или функцию приспособленности:

$$\mathbf{f}(B_M, B_J) = \frac{(B_M \cdot B_J)}{max\{B_M \cdot B_M, B_J \cdot B_J\}}. \tag{3.2}$$

Область значений целевой функции (функции приспособленности): $[0, 1]$. Она равна единице для одинаковых сигналов и близка к нулю для сигналов, которые не похожи друг на друга. Таким образом задача сведена к поиску глобального макисмума целевой функции (задача оптимизации):

$$\max \mathbf{f}(x)$$
$$x \in D.$$

Подготовка данных. Для того, чтобы отчистить сигнал от шума используем фильтр нижних частот с частотой среза 0.01 и количеством коэффициентов 200, а также исключаем тренд. Для того, чтобы уменьшить вычислительную трудоёмкость производится децимация сигнала с коэффициентом прореживания 5. После фильтрации и децимации сигнал имеет

следующий вид 3.2.

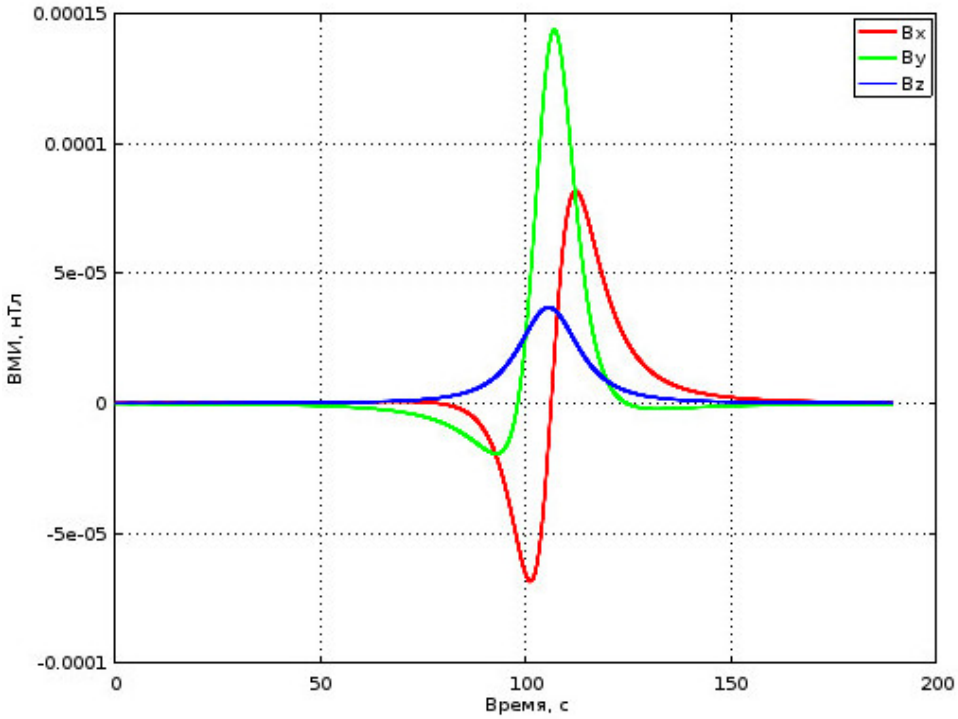

Рис. 3.2: Сигнал показаний векторного магнитометра после фильтрации и децимации.

В качестве сравнения эффективности алгоритмов будем рассматривать точность решения (значение функции приспособленности) и время, затраченное процессором, для выполнения программы.

3.2 Сравнение генетического алгоритма и метода отжига

Параметры генетического алгоритма: максимальное количество итераций алгоритма $N_{max} = 50$, наилучшее значение функции приспособленности $\tilde{f}_{opt} = 0.9$, минимальное изменение приспособленности $\epsilon = 0.001$. Процент количества популяции в процедуре скрещивания и мутации 70% и 20% соответственно. Размер первой популяции — 200 особей, а размер 50-й популяции — 100 особей. Размер популяции равномерно уменьшается с течением времени.

В качестве оценки эффективности алгоритма рассматривается функция приспособленности. Ее зависимость от номера итерации алгоритма показана на рис. 3.3.

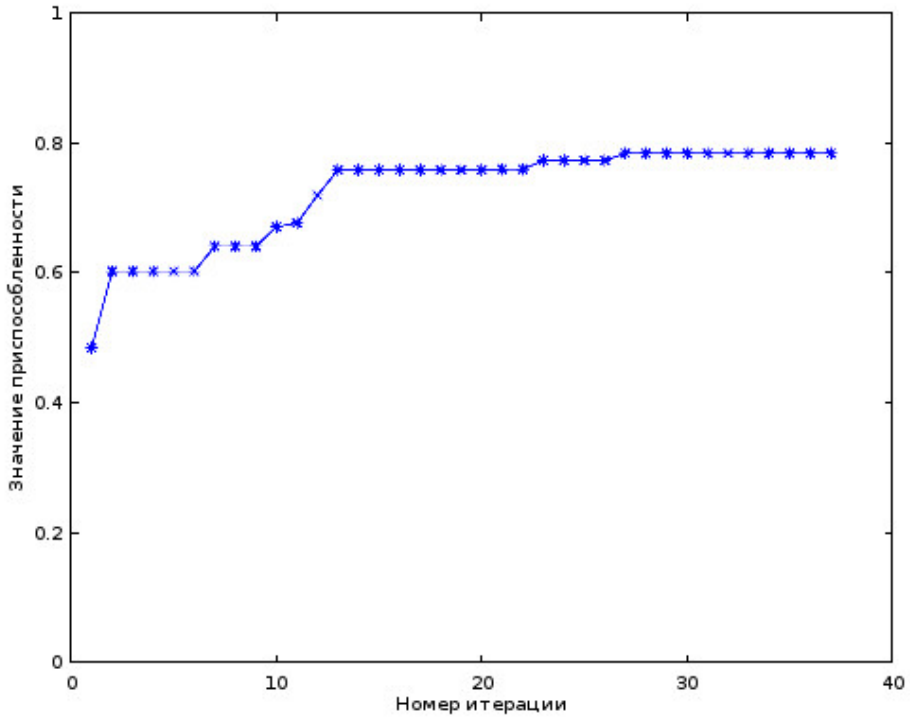

Рис. 3.3: Значение функции приспособленности, генетический алгоритм.

Параметры метода отжига: $N_{max} = 50$, $T = 0.1$, $k = 0.9$ и $N_{neig} = 50$.

Получаем следующие результаты. Оптимальное значение функции подгонки — 0.76.

3.3 Сравнение алгоритма роя частиц и алгоритма гравитационного поиска

Параметры алгоритма роя частиц выбраны следующие: количество итераций алгоритма $N_{max} = 50$, количество частиц в рое $N = 100$, и коэффициенты вклада $\omega = c_1 = c_2 = 0.5$.

Параметры алгоритма гравитационного поиска выбраны следующие: количество итераций алгоритма $N_{max} = 50$, количество частиц $N = 100$.

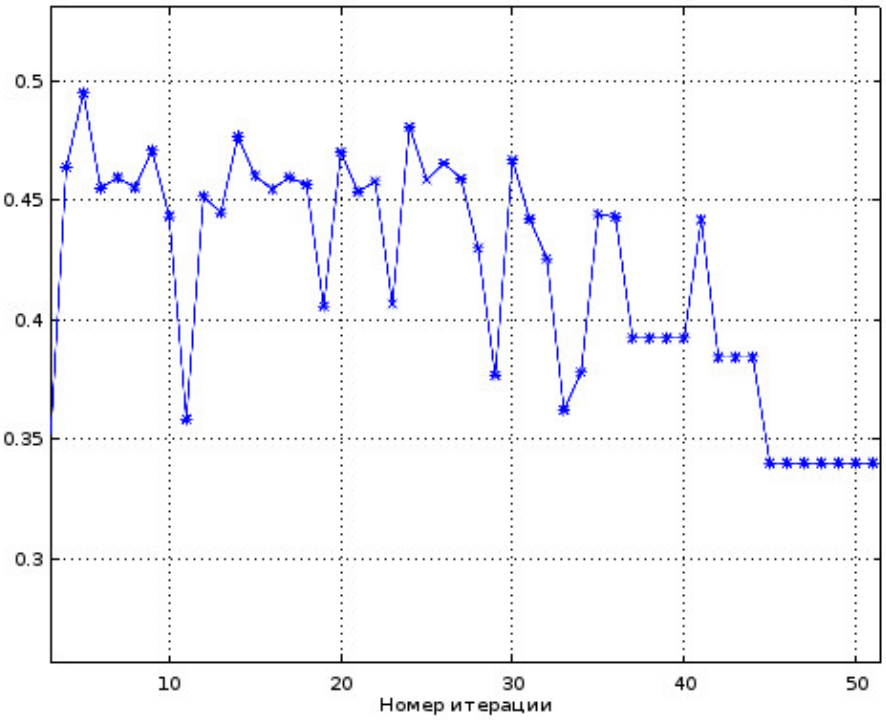

Рис. 3.4: Энергия состояний

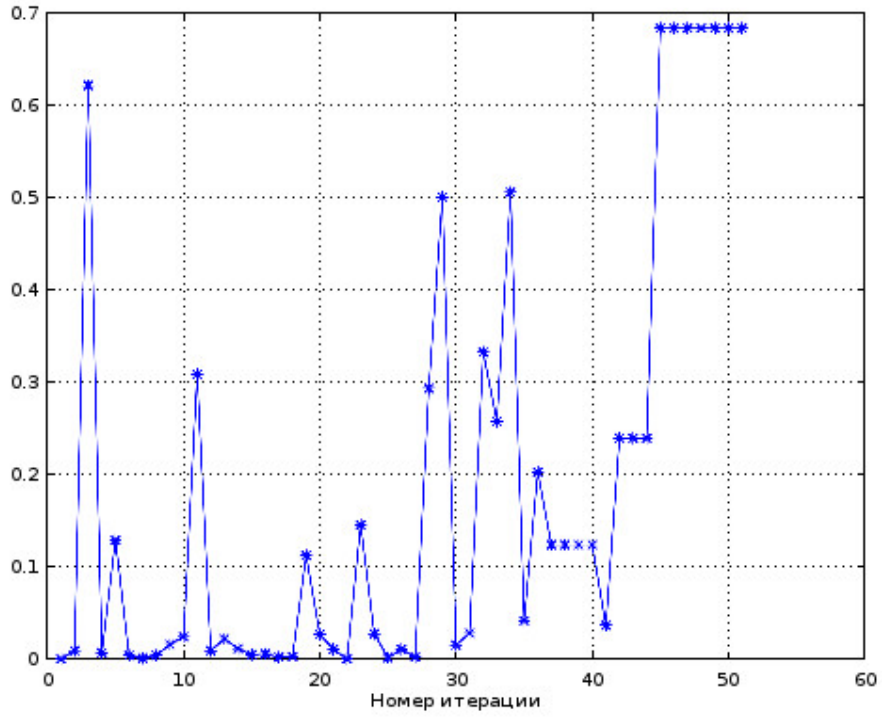

Рис. 3.5: Значение функции подгонки, метод отжига.

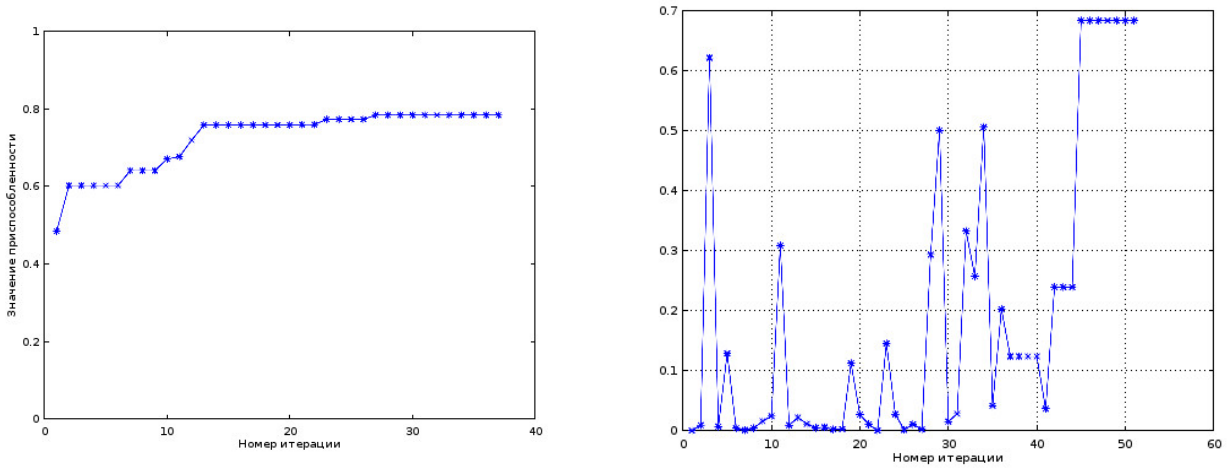

Рис. 3.6: Сравнение функций приспособленности: генетический алгоритм(слева) и метод отжига (справа).

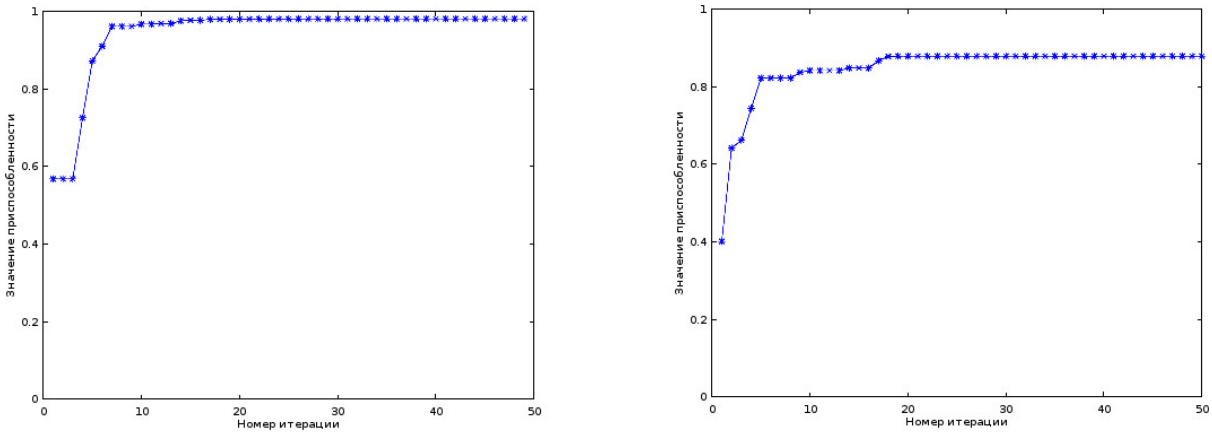

Рис. 3.7: Сравнение функций приспособленности: метод роя частиц (слева) и метод гравитационного поиска (справа).

Параметры c_1 и c_2 выбраны таким образом чтобы обеспечивать хорошее исследование всей области поиска (вклад параметра c_1) и для того чтобы хорошо исследовать область глобального экстремума. В качестве модификации данного метода можно использовать параметры, изменяющиеся с течением времени. Для начальных итераций алгоритма задавать параметр c_1 близким к единице, параметр c_2 близким к нулю, а ближе к последним итерациям, наоборот, задавать параметр c_1 близким к нулю, параметр c_2 близким к единице.

Отметим, что оба алгоритма: метод роя частиц и метод гравитационного поиска основаны на поведении децентрализованной самоорганизующейся системы, однако схема движения частиц в пространстве поиска различается. Если перемещение частицы в методе роя частиц зависит только от лучшего состояния данной частицы и от лучшего глобального состояния всего роя, то в методе гравитационного поиска перемещение частицы зависит от всей системы в целом. Время работы алгоритма гравитационного поиска существенно больше времени работы других алгоритмов, что связано с вычислением на каждой итерации алгоритма гравитационной силы каждых двух частиц. Т. е. получаем квадратичную трудоемкость вместо линейной.

Заметим, что целевая функция имеет малое количество экстремумов, поэтому более простой метод роя частиц оказывается более эффективным, по крайней мере в затратах времени. Для целевой функции, обладающей большим количеством экстремумов, предполагается, что алгоритм гравитационного поиска будет находить более точное решение, так как пространство поиска исследуется лучше.

ВЫВОДЫ

Сравнение всех четырех методов отражено в таблице 3.2.

Таким образом получаем, что наиболее эффективным алгоритмом по всем параметрам является метод роя частиц. Метод гравитационного поиска также показывает очень хорошую точность решения, но значительно уступает по трудоемкости. Генетический алгоритм показал хорошую точность решения и достаточно малую трудоемкость. Метод отжига оказался наименее эффективным.

Хорошие результаты работы алгоритмов, которые отражает значение приспособленности очень близкое к единице, можно объяснить достаточно простой моделью сигнала. Но на практике измеренный магнитометром сигнал содержит намного больше помех, и данная модель может быть не настолько эффективна.

Расчет функции приспособленности основан на вычислении взаимно корреляционной функции, что существенно ускоряет работу программы. Однако использование взаимно корреляционной функции может недостаточно хорошо отражать схожесть сигналов. В качестве альтернативного сравнения двух сигналов можно использовать расстояние Хаусдорфа. Данная метрика в большей степени отражает схожесть сигналов, однако проигрывает по трудоемкости.

Метод	Значение приспособленности (макс)	Время работы программы (сек)
Генетический алгоритм	0.76	16.5
Метод отжига	0.68	13.2
Метод роя частиц	0.98	4.5
Гравитационный поиск	0.88	57

Таблица 3.2: Сравнение эффективности алгоритмов.

.

ЗАКЛЮЧЕНИЕ

Результы работы заключаются в следующем.

1. Написаны программы для четырех эвристических методов, примененных к задаче определения местоположения ферромагнитных объектов.

2. Получены результаты эффективности эвристических алгоритмов:

   (a) Метод роя частиц является наиболее эффективным методом.

   (b) Генетический алгоритм и метод гравитационного поиска имеют хорошие показатели точности решения.

   (c) Метод отжига — наименее эффективный метод среди рассмотренных.

Список литературы

[1] Detection of unexploded ordance (uxo) using marine magnetic gradiometer data / Ahmed Salem, Hamada Toshio, Joseph Kiyoshi Asahina, Keisuke Ushijima // BUTSURI-TANSA(Geophysical Exploration). — 2005. — Vol. 36. — P. 97–103.

[2] Investigation of advanced data processing technique in magnetic anomaly detection systems / B. Ginzburg, L. Frumkis, B. Z. Kaplan et al. // International journal on smart sensing and intelligent systems. — 2008. — Vol. 1. — P. 110–122.

[3] Mcaulay A. Computerized model demonstrating magnetic submarine localization // IEEE Transactions on Aerospace and Electronic Systems AES. — 1977. — Vol. 13. — P. 246–254.

[4] Tian Z. X. Underwater magnetic surveillance system for port protection // IEEE. — 2011.

[5] Magnetic detection of a surface ship by an airborne lts squid mad / M. Hirota, T. Furuse, K. Ebana et al. // IEEE TRANSACTIONS ON APPLIED SUPERCONDUCTIVITY. — 2001. — Vol. 11, no. 1. — P. 884–887.

[6] Merlat L., Naz P. Magnetic localization and identification of vehicles // Proceedings of SPIE. — 2003. — Vol. 5090, no. 174–185.

[7] Ferromagnetic mass localization in check point configuration using a levenberg marquardt algorithm / Roger Alimi, Nir Geron, Eyal Weiss, Tsuriel Ram-Cohen // Sensors. — 2009. — Vol. 9. — P. 8852–8862.

[8] Leach B. W. Aeromagnetic compensation as a linear regression problem // Information linkage between applied mathematics and industry. — 1980. — Vol. 2. — P. 139–161.

[9] Magnetic noise compensation using fir model parameter estimation method / Takayuki Inaba, Akihiro Shima, Masaharu Konishi et al. // Electronics and Communications in Japan (Part III: Fundamental Electronic Science). — 2002. — Vol. 85. — P. 1–11.

[10] Bickel S. H. Small signal compensation of magnetic fields resulting from aircraft maneuvers // IEEE Transactions on aerospace and electronic systems. — 1979. — no. 4. — P. 518–525.

[11] Adaptive cancellation of geomagnetic background noise for magnetic anomaly detection using coherence / Dunge Liu, Xin Xu1, Chao Huang et al. // Measurement Science and Technology. — 2015. — Vol. 26. — 2 Citation.

[12] Energy detection based on undecimated discrete wavelet transform and its application in magnetic anomaly detection / Xinhua Nie, Zhongming Pan, Dasha Zhang et al. // PLOS ONE. — 2014. — Vol. 9.

[13] Levenberg K. A method for the solution of certain non-linear problem in least squares // Q. J. Appl. Math. — 1944.

[14] Marquardt D. An algorithm for least-squares estimation of nonlinear parameter // SIAM Journal on Applied Mathematics. — 1963. — Vol. 11. — P. 431–441.

[15] Murray I. B., McAulay A. D. Magnetic detection and localization using multichannel levinson-durbin algorithm // SIGNAL PROCESSING, SENSOR FUSION, AND TARGET RECOGNITION XIII. — Vol. 5429 of SPIE. — Orlando, FL, 2004. — April. — P. 561–566.

[16] Aeromagnetic search using genetic algorithm / A. Sheinker, N. Salomonski,

B. Ginzburg et al. // Progress In Electromagnetics Research Symposium. — Hangzhou, China, 2005. — August. — P. 492 – 495.

[17] A dedicated genetic algorithm for localization of moving magnetic objects / Roger Alimi, Eyal Weiss, Tsuriel Ram-Cohen et al. // Sensors. — 2015. — Vol. 15. — P. 23788–23804.

[18] Kirkpatrick S., Jr. C. D. G., Vecchi M. P. Optimization by simulated annealing // SCIENCE. — 1983. — Vol. 220, no. 4598. — P. 671–680.

[19] Bertsimas D., Tsitsiklis J. Simulated annealing // Statictical Science. — 1993. — Vol. 8, no. 1. — P. 10–15.

[20] Rashedi E., Nezamabadi-pour H., Saryazdi S. Gsa: A gravitational search algorithm // Information Sciences. — 2009. — Vol. 179. — P. 2232–2248.

[21] Kennedy J., Eberhart R. Particle swarm optimization (pso) // Proc. IEEE International Conference on Neural Networks. — Vol. 4. — 1995. — P. 1942–1948.

[22] Rini D. P., Shamsuddin S. M., Yuhaniz S. S. Particle swarm optimization: Technique, system and challenges // International Journal of Computer Applications. — 2011. — January. — Vol. 14, no. 1. — P. 0975–8887.

[23] Application of target-based and noise-based methods in magnetic anomaly detection systems / B. Ginzburg, A. Sheinker, N. Salomonski et al. // MARELEC - Marine Electromagnetics. — Stockholm, Sweden, 2009. — July.

[24] Wynn W. M. Detection and Identification of Visually Obscured Targets / Ed. by C. E. Baum. — Taylor and Francis, 1999. — P. 337–374.

[25] Rao K. R., Yip P., Rao K. R. Discrete cosine transform: Algorithms, advantages, applications // Academic Press. — Boston, MA, USA, 1990.

[26] Hodrick R. J., Prescott E. C. Postwar u.s. business cycles: An empirical investigation // Journal of money credit and banking. — 1981. — Vol. 29, no. 1.

[27] Localization and magnetic moment estimation of a ferromagnetic target by simulated annealing / Arie Sheinker, Boaz Lerner, Nizan Salomonski et al. // Measurement Science and Technology. — 2007. — Vol. 18. — P. 3451–3457.

Приложение А

Код программ

Представлены основные программы алгоритмов написанные на языке Matlab.

А.1 Генетический алгоритм

```
% координаты магнитометра
sensorCoord1 = [0,0,0]';

% условия критерия останова
N_MAX = 50;
FITNESS_OPT = 0.95;
EPS_DEC = 0.001;

% кол-во хромосом
NUM_GEN_0 = 200; % в начальной популяции
NUM_GEN_N = 100; % в конечной популяции № N_MAX
NUM_GEN = NUM_GEN_0; % в текущей популяции
d_num_gen = round((NUM_GEN_0 - NUM_GEN_N) / N_MAX);

% кол-во наилучших хромосом попадающих сразу в след поколение
```

```
NUM_CHROM_NEXT= 10;
CROSSOVER_P = 0.7; % процент скрещивания /100
MUTATION_P = 0.2; % процент мутации /100

%% чтение данных (записанного сигнала)
isAutoRead = true;
fileNameIn = 's2_filtr.kmm';
[DATA, PathNameIn] = readNaturalData(pwd, fileNameIn, isAutoRead);

% параметры для моделирования поля
timeParam.dt = DATA.dt;
timeParam.startTime = DATA.time(1);
timeParam.finishTime = DATA.time(end);

B_meas = [DATA.tKMx, DATA.tKMy, DATA.tKMz];
% скал произведение B*B
BB_dot = sum(dot(B_meas,B_meas));

initime = cputime;
% область определение генов
alpha_max = pi; % > 0
dist_max = 50; % это расстояние между сенсорами, > 0
velocity_max = 5; % > 0
mag_moment_max = 10; % по модулю

% создание начальной популяции
GEN_DATA.alpha = rand(NUM_GEN,1) * alpha_max;
GEN_DATA.dist = rand(NUM_GEN,1) * dist_max;
GEN_DATA.velocity = rand(NUM_GEN,1) * velocity_max;
GEN_DATA.mx = (rand(NUM_GEN,1) - 0.5) * 2 * mag_moment_max;
GEN_DATA.my = (rand(NUM_GEN,1) - 0.5) * 2 * mag_moment_max;
GEN_DATA.mz = (rand(NUM_GEN,1) - 0.5) * 2 * mag_moment_max;
```

```
for i=1:NUM_GEN
    chromosome(i).alpha = GEN_DATA.alpha(i);
    chromosome(i).dist = GEN_DATA.dist(i);
    chromosome(i).velocity = GEN_DATA.velocity(i);
    chromosome(i).mm_x = GEN_DATA.mx(i);
    chromosome(i).mm_y = GEN_DATA.my(i);
    chromosome(i).mm_z = GEN_DATA.mz(i);
end

n = 1;
bestFitness = zeros(N_MAX, 1);
isStopCondition = 0; % 0 - алг работает, 1-3 алг завершает работу
% главный цикл алгоритма
while ~isStopCondition
  fprintf('iter %d:\n',n);
  % значения приспособленности хромосом
  fitness = zeros(NUM_GEN, 1);

  % вычисление значение приспособленности всех хромосом
  for i=1:NUM_GEN
    fitness(i) = get_fitness(chromosome(i), sensorCoord1, timeParam,
     B_meas, BB_dot);
    chromosome(i).fitness = fitness(i);
  end

  bestFitness(n) = max(fitness);

  % КРИТЕРИЙ ОСТАНОВА
  if n > 10
    isStopCondition = get_isStopCondition(n, N_MAX, FITNESS_OPT, EPS_D
     bestFitness(1:n), 10);
```

```
end

fitness_threshold = sort(fitness)(end - NUM_CHROM_NEXT-1);
% хромосомы, которые не участвуют в изменениях до след поколения
% будут стоять в начале
cur_tmp = 1;
for j = 1:NUM_GEN
  if (chromosome(j).fitness > fitness_threshold) && (j ~= cur_tmp)
    if chromosome(j).fitness > chromosome(cur_tmp).fitness
      tmp = chromosome(cur_tmp);
      chromosome(cur_tmp) = chromosome(j);
      chromosome(j) = tmp;
      cur_tmp = cur_tmp + 1;
    else
      cur_tmp = cur_tmp + 1;
      j = j - 1;
    end
  end
  if cur_tmp > NUM_CHROM_NEXT
    break;
  end
end

for j = 1:NUM_GEN
  fitness(j) = chromosome(j).fitness;
  chromosome_new(j) = chromosome(j);
end
 fprintf('\t max_fitness = %f\n', max(fitness(NUM_CHROM_NEXT+1:end)))

% изменение размера популяции
NUM_GEN_NEW = NUM_GEN - d_num_gen;
```

```
  % СЕЛЕКЦИЯ
  chromosome = selection(chromosome, n, N_MAX, NUM_GEN,
   NUM_GEN_NEW, NUM_CHROM_NEXT);
  NUM_GEN = NUM_GEN_NEW;

  % СКРЕЩИВАНИЕ
  % количество скрещиваемых пар
  num_cross = round(CROSSOVER_P * NUM_GEN/2);
  for j = (NUM_CHROM_NEXT + 1): (NUM_CHROM_NEXT + 1 + 2*num_cross)
    [chromosome(j),  chromosome(j+1)] = crossover(chromosome(j), ...
     chromosome(j+1), sensorCoord1,timeParam, B_meas, BB_dot);
    j = j + 1;
  end
  for j = 1:NUM_GEN
    fitness(j) = chromosome(j).fitness;
  end

  % МУТАЦИЯ
  % количество индивидуумов, участвующих в мутации
  num_mutation = round(MUTATION_P * NUM_GEN);
  for j = (NUM_CHROM_NEXT + 1): (NUM_CHROM_NEXT + 1 + num_mutation)
    chromosome(j) = mutation(chromosome(j), alpha_max, dist_max, ...
     velocity_max, mag_moment_max);
    j = j + 1;
  end
  n = n + 1;
end
fintime = cputime;
```

A.2 Метод отжига

```
% Метод отжига
% координаты магнитометра
sensorCoord = [0,0,0]';

% условия критерия останова алгоритма
N_MAX = 50; % максимальное количество итераций
FITNESS_OPT = 0.85;
EPS_DEC = 0.001;

% Параметры алгоритма
T = 0.1; % начальная температура
cooling_rate = 0.97; % коэф. уменьшения температуры
num_neighboring = 50; % кол-во соседних состояний

%% чтение данных (записанного сигнала)
isAutoRead = true;
fileNameIn = 's2_filtr.kmm';
[DATA, PathNameIn] = readNaturalData(pwd, fileNameIn, isAutoRead);

initime = cputime;

% параметры для моделирования поля
timeParam.dt = DATA.dt;
timeParam.startTime = DATA.time(1);
timeParam.finishTime = DATA.time(end);

B_meas = [DATA.tKMx, DATA.tKMy, DATA.tKMz];
% скал произведение B*B
BB_dot = sum(dot(B_meas,B_meas));
```

```
% область определения неизвестных переменных (параметров модели)
alpha_max = pi; % > 0
dist_max = 40; % это расстояние между сенсорами, > 0
velocity_max = 5; % > 0
mag_moment_max = 10; % по модулю

function chromosome = simulate_chromosome(alpha_max, dist_max,...
 velocity_max, mag_moment_max)
  chromosome.alpha = rand(1) * alpha_max;
  chromosome.dist = rand(1) * dist_max;
  chromosome.velocity = rand(1) * velocity_max;
  chromosome.mm_x = (rand(1) - 0.5) * 2 * mag_moment_max;
  chromosome.mm_y = (rand(1) - 0.5) * 2 * mag_moment_max;
  chromosome.mm_z = (rand(1) - 0.5) * 2 * mag_moment_max;
end;

% создание начальной популяции
chromosome =  simulate_chromosome(alpha_max, dist_max, ...
velocity_max, mag_moment_max);

n = 1;
isStopCondition = 0; % 0 - алг работает, 1-3 алг завершает работу
Energy_iter = zeros(N_MAX, 1);
fitness = zeros(N_MAX, 1);
% вычисляем текущую энергию
chromosome.energy = get_energy(chromosome, sensorCoord, ...
timeParam, B_meas, DATA.N);

% наилучшая хромосома на данный момент
chromosome_opt = chromosome;

% главный цикл алгоритма
```

```
while ~isStopCondition
%  fprintf('iter %d:\n',n);

  % запоминаем энергию в текущей итерации
  Energy_iter(n) = chromosome.energy;
  fitness(n) = get_fitness(chromosome, sensorCoord,timeParam, B_meas, I

  i_cur = 0;
  while i_cur < num_neighboring
    % создаем новое состояние
    chromosomchromosome_next.energy =  simulate_chromosome(alpha_max,
     dist_max, velocity_max, mag_moment_max);

    % считаем энергию в состоянии chromosomchromosome_next.energy
    chromosome_next.energy = get_energy(chromosomchromosome_next.energy
      sensorCoord, timeParam, B_meas, DATA.N);

    % пробуем перейти в состояние chromosomchromosome_next.energy
    if chromosome_next.energy < chromosome.energy
      chromosome = chromosomchromosome_next.energy;
      chromosome.energy = chromosome_next.energy;
    else
      trans_prob = exp((chromosome.energy - chromosome_next.energy)/T)
      if rand(1) < trans_prob % переходим в след. состояние
        chromosome = chromosomchromosome_next.energy;
        chromosome.energy = chromosome_next.energy;
      end
    end
    i_cur = i_cur + 1;
  end

  % Уменьшаем температуру
```

```
  T = cooling_rate * T;

  % запоминаем наилучшую хромосому
  if chromosome_opt.energy > chromosome.energy
    chromosome_opt = chromosome;
  end

  % КРИТЕРИЙ ОСТАНОВА
  if n > 10
    isStopCondition = get_isStopCondition(n, N_MAX, ...
    FITNESS_OPT, EPS_DEC);
  end

  n = n + 1;
end
n = n-1;
fintime = cputime;
```

A.3 Метод роя частиц

```
% метод роя частиц
% условия критерия останова
N_MAX = 50; % кол-во итераций алгоритма
NUM_GEN = 100; % количество индивидуумов в рое

% параметры алгоритма
omega = 0.5;
fi_p = 0.5;
fi_g = 0.5;

%% чтение данных (записанного сигнала)
isAutoRead = true;
fileNameIn = 's2_filtr.kmm';
[DATA, PathNameIn] = readNaturalData(pwd, fileNameIn, isAutoRead);

initime = cputime;
% координаты магнитометра
sensorCoord1 = [0,0,0]';

% параметры для моделирования поля
timeParam.dt = DATA.dt;
timeParam.startTime = DATA.time(1);
timeParam.finishTime = DATA.time(end);

B_meas = [DATA.tKMx, DATA.tKMy, DATA.tKMz];
% скал произведение B*B
BB_dot = sum(dot(B_meas,B_meas));

% область определения генов
alpha_max = pi; % > 0
```

```
dist_max = 40; % это расстояние между сенсорами, > 0
velocity_max = 5; % > 0
mag_moment_max = 5; % по модулю
mod_norm = norm([alpha_max, dist_max, velocity_max, mag_moment_max, ..
mag_moment_max,mag_moment_max]);
NUM_DIM = 6; % количество "генов в хромосоме"

x = zeros(NUM_GEN, NUM_DIM);

% создание начального роя (популяции)
x(:,1) = rand(NUM_GEN,1) * alpha_max;
x(:,2) = rand(NUM_GEN,1) * dist_max;
x(:,3) = rand(NUM_GEN,1) * velocity_max;
x(:,4) = (rand(NUM_GEN,1) - 0.5) * 2 * mag_moment_max;
x(:,5) = (rand(NUM_GEN,1) - 0.5) * 2 * mag_moment_max;
x(:,6) = (rand(NUM_GEN,1) - 0.5) * 2 * mag_moment_max;

p = zeros(NUM_GEN, NUM_DIM);
p = x;
fitness = zeros(NUM_GEN, 1);

% вычисление значение приспособленности всех частиц
for i=1:NUM_GEN
  fitness(i) = get_fitness(x(i,:), sensorCoord1, timeParam,...
   B_meas, BB_dot);
%  particle(i).fitness = fitness(i);
end

fitness_best = zeros(N_MAX,1);
[ fitness_best(1), max_index ] = max(fitness);
g = zeros(N_MAX, NUM_DIM);
g(1,:) = x(max_index,:);
```

```
%velocity = zeros(NUM_GEN, 1);
%velocity = (rand(NUM_GEN,1)*2-1) * mod_norm;

velocity(:,1) = rand(NUM_GEN,1) * alpha_max;
velocity(:,2) = rand(NUM_GEN,1) * dist_max;
velocity(:,3) = rand(NUM_GEN,1) * velocity_max;
velocity(:,4) = (rand(NUM_GEN,1) - 0.5) * 2 * mag_moment_max;
velocity(:,5) = (rand(NUM_GEN,1) - 0.5) * 2 * mag_moment_max;
velocity(:,6) = (rand(NUM_GEN,1) - 0.5) * 2 * mag_moment_max;

n = 1; % текущая итерация алгоритма
while n <  N_MAX
  for i=1:NUM_GEN
      rp = rand(NUM_DIM,1)';
      rg = rand(NUM_DIM,1)';
      v_tmp = omega * velocity(i,:) +fi_p * rp .* (p(i,:) - x(i,:))...
        +fi_g * rg .* (g(n,:) - x(i,:));
      velocity(i,:) = v_tmp;
      x(i,:) = x(i,:) + velocity(i,:);

      fitness_tmp = get_fitness(x(i,:), sensorCoord1, timeParam,...
       B_meas, BB_dot);
      if fitness_tmp > fitness(i)
        fitness(i) = fitness_tmp;
        p(i,:) = x(i,:);
        if fitness_tmp > fitness_best(n)
          fitness_best(n) = fitness_tmp;
          g(n,:) = p(i,:);
        end
      end
  end
```

```
  n = n + 1;
  fitness_best(n) = fitness_best(n-1);
  g(n,:) = g(n-1,:);
end
fintime = cputime;
```

A.4 Метод гравитационного поиска

```
% Метод гравитационного поиска
function [Fbest,Lbest,BestChart,MeanChart,req_time]=GSA(F_index,N,...
max_it,ElitistCheck,min_flag,Rpower)

if F_index == 0
  %% чтение данных (записанного сигнала)
  isAutoRead = true;
  fileNameIn = 's2_filtr.kmm';
  [DATA, PathNameIn] = readNaturalData(pwd, fileNameIn, isAutoRead);

initime = cputime;
Rnorm=2;
% ограничения области поиска
[low,up,dim]=test_functions_range();
% инициализация
X=initialization(dim,N,up,low);
BestChart=[];MeanChart=[];
V=zeros(N,dim);

for iteration=1:max_it
    % проверка на принадлежность области
    X=space_bound(X,up,low);
    fitness=evaluateF(X, DATA);
    if min_flag==1
    [best best_X]=min(fitness);
    else
    [best best_X]=max(fitness);
    end
    if iteration==1
       Fbest=best;Lbest=X(best_X,:);
```

```
    end
    if min_flag==1
      if best<Fbest  %minimization.
       Fbest=best;Lbest=X(best_X,:);
      end
    else
      if best>Fbest  %maximization
       Fbest=best;Lbest=X(best_X,:);
      end
    end

BestChart=[BestChart Fbest];
MeanChart=[MeanChart mean(fitness)];

% Вычисление масс
[M]=massCalculation(fitness,min_flag);
 G=Gconstant(iteration,max_it);
 % Вычисление ускорения
 a=Gfield(M,X,G,Rnorm,Rpower,ElitistCheck,iteration,max_it);
 [X,V]=move(X,a,V);
end
fintime = cputime;
```